Performance Data Usage Scenarios (Draft 1)

Grid Forum Performance Working Group

Last Updated: October 2, 2000

Compiled by Ruth Aydt and Darcy Quesnel

History and Purpose of Document:

At the Grid Forum 4 meeting, the Performance Working Group decided it would be helpful to collect a set of scenarios outlining how people envisioned using performance data in the Grid environment. Based on our ongoing discussions, it seemed clear that most members of the working group had one or more ideas about how performance data would be used, but these "stories" were never written down. It was felt that collecting and transcribing the scenarios would give us a common set of "target consumers" that could be used to evaluate proposed architectures, schema, taxonomy, and tools.

Working Group members were solicited for scenarios in any format. No restrictions were placed on level of detail (some relate very specific usage scenarios while others outline classes of consumers) or feasibility (some can be done today, others are only a dream at this point).

The collected scenarios appear in this document, with numbers assigned to each. Following the text of each scenario are two additional sections, "Bullet Summary", and "Comments / Deductions / Assumptions / Questions". The bullet summary is an initial attempt to tersely restate the scenario capturing the important points. The comments/deductions/assumptions/questions section contains one reader's "stream of consciousness" prompted by the scenario. These include implications of the scenario on the grid performance analysis environment, questions raised but not answered by the scenario text, some initial thoughts on grid-specific issues raised by the scenario, and in general other miscellaneous things. Note that statements in the bullet and comment sections have not been reviewed/verified by the scenario authors and could be inaccurate in this version of the draft.

As this living document matures, we hope to refine and "flesh out" each scenario so that it clearly describes to everyone in the working group the intentions of the scenario author. Hopefully some standard format for expressing the scenario requirements will emerge so that the group can use these scenarios (and new ones) to evaluate the feasibility of the proposed architectures/standards/tools/taxonomies in relationship to the needs of the data consumer. Eventually we may add details as to how the scenario would be (or has been) implemented in the environment we propose.

The initial draft will be discussed at Grid Forum 5 … please come prepared to contribute your insights. Scenario authors – please review and comment! We also welcome feedback by email. Send comments to: aydt@cs.uiuc.edu or darcy.quesnel@anl.gov Please contribute… we need input from as many people as possible if this document is to be a useful resource for the group.

Scenario 1

Text:
The developer instruments a library (say, Globus I/O) to produce events with some information (say, bandwidth in MB/s). For performance and thread compatibility reasons, the developer doesn't want consumers to be able to subscribe to the instrumented application - the developer just wants to be able to produce events and forget about them. So the developer leaves it up to the user of the library to specify as an argument where the events should go.

The user of an application (say, VolViz) linked to the instrumented library gives the address of either a consumer or a registry of consumers when invoking the application that is passed on to the library. In this case the user doesn't know the address of a specific consumer but does know the address of a registry. The user wants a consumer that makes events available through a producer interface as soon as it consumes them. So the user includes these attributes that must be met by a consumer when specifying the address of the registry to the application.

The system manager wants to monitor the performance of an application (say, VolViz) while it's running. The manager starts a simple stripchart visualization tool that acts as a consumer of events and which accepts as an argument either the address of a producer or a registry of producers. In this case the manager doesn't know the specific address of a consumer but does know the address of a registry. The manager wants producers that are producing event information (say, bandwidth in MB/s) from a particular application (say, VolViz). So the manager includes these attributes that must be met by a producer when specifying the address of the registry to the tool.

Bullet Summary:

- Developer instruments a library to produce events
- Developer willing to send events somewhere but not willing to let user subscribe to them dynamically directly from the application using instrumented library
- User controls what tasks should be logged via a command line flag, and specifies where log data should be sent – either to a consumer or to a registry.
- User has some control over characteristics of a consumer that may get the data and specifies those via attributes when application is started.
- Consumer of data can query producer or registry and match events with specific attributes.

Comments / Deductions / Assumptions / Questions:

- Either there must be agreement between producers and consumers about data format or there must be some exchange that lets the format be "discovered".
- Producer pushes data in this model – consumer has no control over when it gets it or how much it receives
- Unsure what this sentence means exactly: *The user wants a consumer that makes events available through a producer interface as soon as it consumes them.* What is the producer interface and is it used in the VolViz example?
- Some Grid-specific issues related to this scenario are:
    - Where will registry be located and will there be multiple registries?
    - How does the consumer know where to look for the registry?
    - If there is a large volume of data and the registry is "far away" and the consumer is "near" is there a way to reduce network delays as the data is copied there and back?
    - Will the registry introduce a limiting factor for time-sensitive data?
    - If data is not in text format and 2 consumers get the same data from a single registry how will they get it in correct byte order?

Scenario 2

Text:
NetLogger/DPSS: The DPSS (a distributed system) clients/servers are instrumented to produce NetLogger events. When a user performs a task using a DPSS client, they can pass a command-line flag to indicate that logging should be performed for the duration of the task (a file transfer, in this case), and additional arguments to indicate where the log data should be sent. As part of NetLogger, a collection daemon can run on some host that receives streams of events from a number of sources and writes them to a log file; the user starts this daemon by hand (if not already running) and then specifies its address on the command-line.

Bullet Summary:
- User starts collection daemon by hand
- When application is started, address of log host is specified on command line.
- User controls what tasks should be logged via a command line flag, and specifies where log data should be sent.

Comments / Deductions / Assumptions / Questions:
- This scenario only involves logging data at the application (DPSS) level.
- There is no need to correlate this logged data with grid resource data.
- User is responsible for controlling when and where data is logged, and for starting collection daemon.
- Only consumer of data is the user who collected it so general format is not important. Also, no need for others to discover that this data is being collected.
- Some Grid-specific issues related to this scenario are:

- It may not be feasible for the user to be able to start the logging daemon on a particular host – that may depend on the load on the Grid and the access that user has to the host.
- If this data is logged in real-time and the volume is large, logging to a host "far away" may not be practical.

Scenario 3

Text:
JAMM: Java programs on each host in a distributed system manage a group of "sensors", which poll the system and produce events periodically. A GUI can be used to view which events are available and to subscribe to these events. The user can specify which host in the system receives the subscribed data. Starting/stopping "sensors" can also be done with a separate GUI.

Bullet Summary:
- JAMM is a sensor manager that runs on a host and controls system sensors on that host.
- A GUI lets a user list available events and subscribe to them.
- A separate GUI allows sensors to be started and stopped.

Comments / Deductions / Assumptions / Questions:
- What receives the events on the host that they are directed to?
- What type of information do the events contain and how are they used?
- Some Grid-specific issues related to this scenario are:
  - Who can control the sensors and view the events?
  - How are events from different hosts coordinated into a "grid view"?
  - If the data produced is resource information and not application specific, can it be shared with other consumers who would want this data?

Scenario 4

Text:
JAMM/Port monitor:
A Java program running on a host in the distributed system periodically performs a *netstat* to check for port activity. If a port that it is configured to respond to shows a change in activity, system "sensors" are started (stopped) to produce events. The user can configure the "port monitor", as it is called, to start/stop desired sensors when a given application port (e.g. http, ftp, telnet ports) show changes in activity, thus automatically monitoring the application.

Bullet Summary:
- The port monitor tool enables and disables sensors associated with a given port.
- The port monitor can be configured by a user to start/stop sensors when an application port shows changes in activity.

Comments / Deductions / Assumptions / Questions:

- This scenario seems to focus on the producer of monitoring data rather than a consumer. Who consumes this data?
- Are the sensors pre-determined or somehow specified by the user who configures them?
- What constitutes a change in activity on a port?
- How do you know the change in activity is associated with the application that enabled the monitor and not some other application?
- Some Grid-specific issues related to this scenario are:
    - How do you make sure only one user tries to configure a port monitor for a given port?
    - Is the amount of sensor data somehow bounded or could it flood the environment?
    - Where does the sensor data go? How is it discovered or propagated?
    - Who starts the java program on the host initially and will all grid systems allow such a program?

Scenario 5

Text:
Fault detection and analysis: monitoring data can be used to determine if a job is still running, or has died or hung. Monitoring data can also be used to help determine the cause of the fault.

Bullet Summary:

- Did a fault occur and if so, where/why.

Comments / Deductions / Assumptions / Questions:

- See scenario 6 for more discussion on job status.
- Seems some of the needed monitoring data would be job-specific (progress) while others would be resource-specific (network down).
- Some Grid-specific issues related to this scenario are:
    - How to correlate a detected fault somewhere in the Grid with a specific job?
    - Can this data be used not just on a per-job basis but by a "larger consumer"? Maybe like a grid health monitor?

Scenario 6

Text:
Job progress monitoring: users of long running jobs often want to query for the current status of their job. If the job produced the right level of monitoring data, the status could be determined. Job status information could also be graphed, analyzed, or archived.

Bullet Summary:

- Need to be able to monitor long-running jobs.

Comments / Deductions / Assumptions / Questions:

- Would it be up to the "author" of the job to insert the monitors? Would it be feasible to monitor at a higher level?
- If a library or toolkit for inserting job monitor probes could be developed then a standard format could be used and tools developed to work with that.
- Who would be able to monitor the job progress?
- Some Grid-specific issues related to this scenario are:
    - Where would the job monitors report since the job is running in a distributed environment?
    - Should progress be reported on job overall or on "elements" of the job running in different Grid resources?

Scenario 7

Text:
Detailed performance analysis: determining the performance bottlenecks in a complex distributed system requires a large amount of precision time-stamped monitoring data.

Bullet Summary:

- Locating performance bottlenecks requires lots of data from many sources with accurate timestamps.

Comments / Deductions / Assumptions / Questions:

- Monitoring data from many different resources must be viewed in combination to locate bottlenecks – this requires accurate and precise timestamps.
- The scenario doesn't specify that this needs to be done in real-time but with the dynamic nature of Grid resource availability, post-mortem analysis may not help at all with the next execution. It could help post-mortem for resource planning.
- Will this analysis be performed by a user or by a tool?
- How will the bottleneck detector determine which resources are used by the application being analyzed?
- How to minimize impact of collection and distribution of the monitoring data on the behavior of the application being studied?
- Maybe it's not an application that's being studied but rather the Grid resources as a system overall.
- Some Grid-specific issues related to this scenario are:
    - Most issues related to this issue are complicated by, if not directly caused by, the nature of the Grid.

Scenario 8

Text:
Network-aware applications: with access to the right network monitoring information, applications could be self-tuning or self-configuring, modifying themselves to optimally use the available resources.

Bullet Summary:

- Availability of network performance information would allow an application to be self-tuning.

Comments / Deductions / Assumptions / Questions:

- What would be "the right" network monitoring information?
- Here it seems the application would adapt based only on the network information but not on the performance of other resources.
- How fresh would the data need to be in order to be useful?
- Perhaps time window summary data would be better than discreet measurements for this purpose.
- How would this differ from data needed by schedulers?
- Some Grid-specific issues related to this scenario are:
    - If many consumers of this data, how to optimize capture and distribution?
    - Privacy of network usage data
    - Data format understood by application

Scenario 9

Text:
Data replication services: monitoring data is required by data replication services to be able to automatically migrate data to the "best" location.

Bullet Summary:

- Data may be replicated to optimize access time – this requires monitoring data

Comments / Deductions / Assumptions / Questions:

- What sort of monitoring data would be required by these services? Perhaps network, location of consumer process, I/O server load?
- Requires some knowledge of where data can be placed – are all I/O servers reported on an ongoing basis or does the data replicator query for monitoring data only when it considers performing a migration?
- Is the data migrated or replicated? If replicated are there consistency issues (must we monitor which is stale?)
- At what level are the monitors that realize a process is requesting data inserted?
- Some Grid-specific issues related to this scenario are

- If large datasets are being migrated should the data replication service somehow notify in advance (see scenario 10) that it is going to use a lot of network bandwidth or do the resource monitors for the network just "notice" that as the network utilization goes up?

Scenario 10

Text:
Scheduling and prediction services: monitoring data is required by grid scheduling systems to determine the optimal resources to use for a given job.

Bullet Summary:
- Schedulers need resource usage information

Comments / Deductions / Assumptions / Questions:
- It is critical that this information be fresh
- Should it be collected in a common location or distributed across the resources and "pulled" from them when needed by the scheduler?
- Some Grid-specific issues related to this scenario are
  - Privacy/Security issues related to information
  - Varying delays in time to deliver the information based on network conditions

Scenario 11

Text:
Auditing systems: monitoring data is required for accounting purposes, to verify resource utilization, and to verify you are getting the level of service you are paying for.

Bullet Summary:
- Resource usage records required for accounting/auditing.

Comments / Deductions / Assumptions / Questions:
- It seems both the accountants and the users would be accessing the data in the scenario described.
- Implies resource utilization must be collected and reported on a per user (or per $-account) basis.
- When during execution is this data collected? Are there periodic snapshots of the usage profile on each resource or accumulated totals associated with each job? If the job migrates and accumulated per job, the job must "notice" when it migrates.
- Some Grid-specific issues related to this scenario are
  - Privacy/Security issues related to information
  - Where would this auditing /accounting take place as users would presumably have access to systems under the management of very diverse groups?

- Trust – who will you trust and how will you in fact verify that numbers are accurate?

Scenario 12

Text:
Configuration monitoring: a monitoring system could be used to automatically maintain databases of current hardware and software resources. Most current resource databases are generated by hand, are almost always out of date. Using real-time monitoring data to update the database ensures that the databases are accurate.

Bullet Summary:
- Hardware and software resource status saved in database and updated in real-time

Comments / Deductions / Assumptions / Questions:
- How does the hardware and software initially get entered into the database? Is it preloaded or "discovered"?
- How close to real-time would be acceptable update frequency? Would this depend on the particular resource?
- Some Grid-specific issues related to this scenario are
  - Privacy/Security issues related to information in database
  - Can this database serve as a source of this information for many consumers and thereby lessen the load on the producers of the data?

Scenario 13

Text:
A user instruments his/her application using some type of tool (such as Pablo, AIMS or Jumpshot) to collect performance data during the execution of the application. The data is saved to a file for later access. This data is used to performance tune the applications (based upon knowledge gained from reviewing the archived performance data timed events) or develop performance models. In addition to being archived to a file, the application performance data can be used by the application for steering (e.g., use alterative solvers or algorithms or migrate tasks to a better machine).

Bullet Summary:
- Application instrumentation via a tool under control of user
- Data is collected as application executes and logged to a file for later use
- User (or automated process) later uses archived performance data to either
  - Tune the application
  - Develop performance models
- Data can also be used by the application (in real time) for steering – guide selection of optimal solvers or trigger task migration to better machine

Comments / Deductions / Assumptions / Questions:

- User (or automated tool) will be aware of the format of the data produced by the application instrumentation. Therefore, common format not critical for this scenario.
- For data collection and archival, there is no pressure to move the data quickly over the network to a remote system.
- For real-time steering, nothing is explicitly stated in the text but I believe there is an assumption that machine performance data will be used in conjunction with the application data to make steering decisions ("better machine"). This means the application performance data and the machine data needs to be somehow coordinated.
- For real-time steering, possibly a subset or summary of all collected data is all that is necessary. If that is true, only that smaller volume of data needs to be moved across the network in near real-time.
- In this scenario, the only consumer of the data is the user who instruments the application. The data need not be visible to other Grid users or managers.
- Some Grid-specific issues related to this scenario are
    - Collection of trace file(s) into common place and/or archive
    - Coordination of application data with system data
    - Data volume associated with steering requirements

Scenario 14

Text:

An application registers with some given resource monitors (networks, processors, I/O processors, etc.) and uses information from the resource monitor to identify more appropriate algorithms (e.g., use better encoding if working with video or audio) or change some aspects about the applications (e.g., use of more tolerate algorithms) or better schedule the application tasks.

Bullet Summary:

- Application is consumer of performance data
- Performance Data producers are associated with resources on the Grid (networks, processors, I/O engines, etc)
- Data is used for application configuration and scheduling.

Comments / Deductions / Assumptions / Questions:

- As written, the text is not explicit as to whether the application uses the performance data only once, prior to startup, or on an ongoing basis for continuous adaptation throughout the execution.
- If prior to application launch, then one-time "snapshot" of resource status is sufficient. This data should be fresh for things that change (percent idle for

example) and can be fairly stable for things like processor type (which might dictate algorithm choice).

- If ongoing "consumption" of data, then who decides when updates occur? Does the application pull the data from the producers or do the producers push it periodically or when it changes by a given amount? Are summaries (time averages) or predictions needed?
- This data will need to be in a well-understood or well-published format.
- This scenario assumes that the Application will handle the scheduling of the application tasks, not that there will be some master scheduler performing that role based on a requirements list.
- Some Grid-specific issues related to this scenario are
    - How application will discover and connect to producers that are relevant to it.
    - If ongoing monitoring, how to insure fresh data
    - How to avoid flooding system with "point to point" connections between producers and consumers if many consumers want same data.
    - Privacy / Security of "resource" data from many sites consumed by application

Scenario 15

Text:
A top meteorologist submits a simulation / rendering job to predict hurricane behavior via cell phone interface from sailboat in Caribbean, specifying required completion time. Grid scheduler profiles job requirements, locates and reserves appropriate resources, attaching QOS parameters. Job executes on 15 heterogeneous computational platforms around the world, pulling in latest satellite image data, current measurements from various weather sensing devices, and historical data on behavior of previous storms. Execution adapts when one of the systems goes off-line due to high winds knocking out power. CNN broadcasts hurricane warnings, complete with real-time animation from web feed showing the path of the storm. Simulation and visuals adjust continuously based on latest sensor and satellite images. Orderly evacuation is carried out. Meteorologist vacations in Europe next year and leaves cell phone at home.

Bullet Summary:

- Possible Grid Scenario of the future.
- Job submitted with hard deadline.
- Scheduling requires compute, network, data and remote sensor resources.
- Fault detection when system goes off line triggers reassignment of task.
- Strict network requirements for real time display of hurricane path.

Comments / Deductions / Assumptions / Questions:

- Scheduling system could make use of application performance characteristics from previous runs to determines resource needs for this run given time constraints.

- Ongoing monitoring of progress would help adapt if completion time is slipping.
- High priority jobs could be given special privileges, not only for the application itself, but also for the application (and related resource) performance data.
- Real time monitoring would be critical.
- Need to track where job is scheduled and all the resources impacting the job progress. Similar in some ways to performance tuning (scenario 7) but with a possible solution here of allocating more resources to the application.
- Application could have built-in progress checks (see scenario 6) that could be monitored by the scheduling system rather than by the user. And, if not meeting expected progress deadlines, application could adapt.
- Some Grid-specific issues related to this scenario are
    - Difficulty in correlating all the measurements required to pull this scenario off